# Linux下cmake生成SO库的使用实例

1、创建我的共享库：MySharedLib

## CMakeLists.txt

cmake\_minimum\_required(VERSION 2.8)

project(MySharedLib)

# C++11 编译

set(CMAKE\_CXX\_STANDARD 11)

#创建共享库

add\_library(MySharedLib SHARED library.cpp library.h)

## library.h

#ifndef MYSHAREDLIB\_LIBRARY\_H

#define MYSHAREDLIB\_LIBRARY\_H

// 打印 Hello World!

void hello();

// 使用可变模版参数求和

template <typename T>

T sum(T t)

{

return t;

}

template <typename T, typename ...Types>

T sum(T first, Types ... rest)

{

return first + sum<T>(rest...);

}

#endif

## library.cpp

#include <iostream>

#include "library.h"

void hello() {

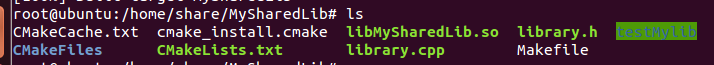
std::cout << "Hello, World!" << std::endl;

}

在CMakeLists.txt所在的目录中：

cmake ./

make



## so 共享库的使用

（被可执行项目调用）创建一个名为test的可执行项目

CMakeLists.txt

#cmake版本

cmake\_minimum\_required(VERSION 2.8)

#项目名：test

project(test)

# C++11 编译

set(CMAKE\_CXX\_STANDARD 11)

#指定生成的版本

#set(CMAKE\_BUILD\_TYPE DEBUG)

#指定编译选项

#set(CMAKE\_CXX\_FLAGS\_DEBUG "-g -Wall")

# 指定头文件路径

set(INC\_DIR /home/share/MySharedLib)

# 库文件路径 指定静态和动态文件目录

set(LIB\_DIR /home/share/MySharedLib)

#指定头文件目录

include\_directories(${INC\_DIR})

#链接库路径

link\_directories(${LIB\_DIR})

#链接库，MySharedLib为库名

link\_libraries(MySharedLib)

#生成目标文件

add\_executable(test main.cpp)

#指定源代码

#set(SOURCE\_FILES main.cpp)

#add\_executable(BoostCoroutineDemo ${SOURCE\_FILES})

# 链接 库文件 MySharedLib 库

target\_link\_libraries(test MySharedLib)

#target\_link\_libraries(BoostCoroutineDemo libboost\_system.a libboost\_thread.a)

## main.cpp

#include <iostream>

#include "library.h"

using std::cout;

using std::endl;

int main() {

hello();

cout << "1 + 2 = " << sum(1,2) << endl;

cout << "1 + 2 + 3 = " << sum(1,2,3) << endl;

return 0;

}

cd testMylib/

cmake ./

make

./test

